Theoretical Computer Science 35 (1985) 191-214
North-Holland

AN INTRODUCTION TO FIFO NETS— MONOGENEOUS NETS: A SUBCLASS OF FIFO NETS*

G. MEMMI**
Ecole Nationale Supérieure des Télécommunications, 46, rue Barrault, 75634 Paris Cedex 13, France

A. FINKEL
LRI and Université de Paris-Sud, Centre d'Orsay, Bât. 490, 91405 Orsay, France

Communicated by M. Nivat
Received April 1983
Revised June 1984

Abstract. We introduce a new model of parallel computation, the FIFO nets. We show how it can simulate Petri nets and coloured Petri nets and prove that a restriction of it (alphabetical FIFO nets) has the power of Turing machines. Furthermore, we define monogeneous FIFO nets and use the coverability graph for proving that it is decidable whether or not a monogeneous net is bounded and whether or not its language is regular.

Key words. FIFO nets, parallel computation model, program machines, regular languages, monogeneous net, boundedness, coverability graph, deterministic languages of Petri nets.

General overview

Of all communication mechanisms between processes of real time systems, two of them hold key positions. One is the mechanism of 'rendez-vous' widely studied with CSP [8] and used in ADA. The other one is the FIFO channel currently used in switching networks, and in languages like SDL and CHILL.

Our aim is to design an operational and formal model well suited to the specification and analysis of sequential processes communicating by FIFO channels.

Beginning with Petri nets [3], we can model each sequential process by a state machine (and states machines form a specific class of Petri nets). Following an idea of Reisig [16], modelling each FIFO channel by a subgraph of a Petri net needs to know its length and leads to rather tedious manipulations. While a powerful abbreviation (in the descriptive sense) could, in principle, be introduced, the problem of modelling a FIFO channel without the knowledge of its length still arises.

* This paper is an extended version of the 6th GI Conf. on Theoretical Computer Science [6].

** This work has been done while the first author was at Thomson CSF-LCR, Domaine de Corbeville, Orsay, France.

This led to the new model of parallel computation to be presented here, the FIFO nets. Again, we begin with Petri nets, but we distinguish tokens in order to model different types of messages (this idea can be found in [17]). Firing rules are then revised so that the model fits in the FIFO mechanism.

Having defined FIFO nets, we show that both Petri and coloured Petri nets can be effectively simulated by them. We then restrict our attention to alphabetical FIFO nets (a subclass of FIFO nets). We prove that a FIFO net can be effectively simulated by an alphabetical one and that an alphabetical FIFO net can be given simulating a given program machine [14], thus showing that FIFO nets have the computational power of Turing machines.

This last result is rather negative with regard to our purpose of analyzing concurrent systems. The main idea of Section 3 is to take advantage of the relationship between Petri nets and our model to extend some known result on Petri nets to a class of FIFO nets. This leads to the construction of monogeneous FIFO nets which contains Petri nets and for which the coverability tree [10] gives a procedure for deciding whether a monogeneous FIFO net is bounded or not. Moreover, we introduce the coverability graph, as in [21], which allows to decide exactly which fifos are bounded.

Finally, we compare monogeneous FIFO net languages with Petri net languages. Our main result is the proper inclusion of the class of Petri net languages in the class of monogeneous FIFO net languages, and the inclusion of this class in the class of deterministic Petri net λ-free languages. We deduce that Petri net labelled (λ-free or not) languages are equal to the monogeneous FIFO nets labelled (λ-free or not) languages; and, also, that regularity is decidable for the monogeneous FIFO net languages.

1. Introduction to FIFO nets

1.1. An example

When searching to model and analyze sequential processes communicating by FIFO channels by using Petri nets, a first approach should be to relax the FIFO mechanism of the channel. Doing so, we authorize some behaviours actually not in the system we model: we can find deadlocks that do not exist. But, and this is more serious, we can have a deadlock in the actual system which vanishes in the Petri net. Let us give an example (from [12]), describing roughly two processes inside a switching system.

As shown in Fig. 1, the process LA is dedicated to a subscriber and the process A to the management of telephonic services. We only show the communication between A and LA, during an On-Hook from the subscriber (modelled by the transitions t_2, t_4, t_7), or a release from A (transition t_{10}). t_1 and t_6 represent an Off-Hook of the subscriber.

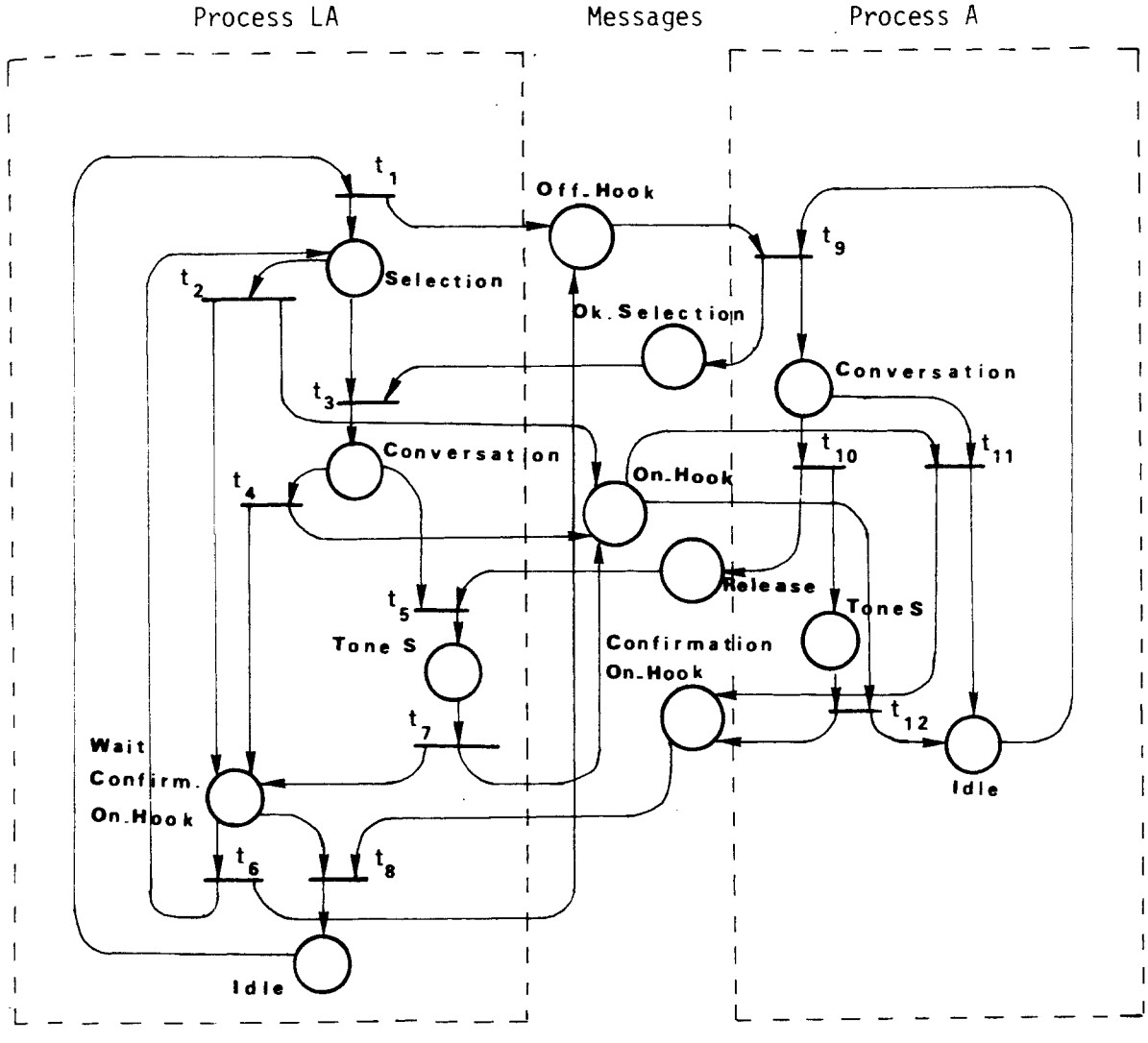


Fig. 1.

Between processes LA and A we find one place per kind of message. The initial marking of this net is defined by one token in the place 'Idle' of each process. It is clear that this Petri net is live. The communication of the two processes is actually implemented by two FIFO channels: F_1 for the messages Off-Hook and On-Hook sent by LA to A; F_2 for the messages OK-Selection, Release, Confirmation On-Hook sent by A to LA.

After the firing of the sequence $t_1\ t_9\ t_3\ t_{10}\ t_4$, the message Release locks the head of F_2; then LA can only fire the sequence $t_2\ t_6$ infinitely often.

This kind of misfunctioning shows the necessity to take into account the FIFO mechanism. The fact that two FIFO channels are not bounded leads us to forgive the design of some abbreviation for a subnet modelling a FIFO. Finally, we design a new model.

1.2. Basic definitions

Let us recall some notations in use in the theory of formal languages. Given any finite set A, A^* denotes the free monoid with base A. The empty word is denoted

by λ; $|x|$ is the length of the word x of A^*. $x \leqslant y$ means that there exists a z of A^* such that $y = xz$; we say that x is a *prefix* of y.

Let $\omega = \text{card}\,\mathbb{N}$ and A^ω the set of infinite words on A. We have $A^\infty = A^* \cup A^\omega$.

A FIFO net is a triplet $\langle R ; M_0 ; A\rangle$ where R is a finite valuated bipartite graph. R describes the topology of the net as in a Petri net [3]; M_0 is the initial marking which defines the initial state of the net; A is a finite alphabet associated to the messages. Formally, we have the following definition.

Definition. A *FIFO net* is a triplet $\langle R ; M_0 ; A\rangle$ where:

(i) $R = \langle F, T ; \Gamma ; V\rangle$ is a finite valuated bipartite graph where F is the set of *fifo* queues (shortly: fifo) (represented by circles), T the set of *transitions* (represented by bars) $(F \cap T = \emptyset)$; Γ is the *corresponding* between F and T giving for each node the set of its successors: V is a valuation on A^*; V is a function from $(T \times F) \cup (F \times T)$ into A^* with $V(x, y) = \lambda$ iff $y \notin \Gamma(x)$.

(ii) M_0, the *initial marking*, is a function from F to A^*.

(iii) A is a finite alphabet. Each letter is called a *message*.

A FIFO net is often shortly denoted by $\langle R ; M_0\rangle$ without mentioning the name of the alphabet.

Now, we can describe our first example taking into account the FIFO channels F_1 and F_2 (see Fig. 2).

F_1 is obtained by folding Off-Hook and On-Hook, F_2 by folding OK-Selection, Release, Confirmation On-Hook.

As in a Petri net or any transition system, the marking of a FIFO net changes by firing transitions.

Definition. A transition t of T is *enabled* at the marking M iff $\forall f \in F$: $V(f, t) \leqslant M(f)$; t enabled at M is denoted by $M\langle t\rangle$.

g is the partial function from $A^* \times A^*$ into A^* such that

$$g(a, x) = \begin{cases} y & \text{iff } x = ay, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

$g(a, x)$ is a function that removes a in the word x iff a is prefix of x.

The firing of a transition creates a new marking M' from M with the following rule (where $(A^*)^F$ denotes the set of the functions from F into A^*).

Definition. The *firing* of a transition t is a partial function δ_t from $(A^*)^F$ into $(A^*)^F$ such that

$$\forall M \in (A^*)^F \quad \forall f \in F: \quad \delta_t(M)(f) = g(V(f, t), M(f)) \cdot V(t, f).$$

$\delta_t(M)$ is defined iff $\delta_t(M)(f)$ is defined for any f of F.

$M' = \delta_t(M)$ is denoted by $M\langle t\rangle M'$. We have $M\langle t\rangle$ iff $\delta_t(M)$ is defined.

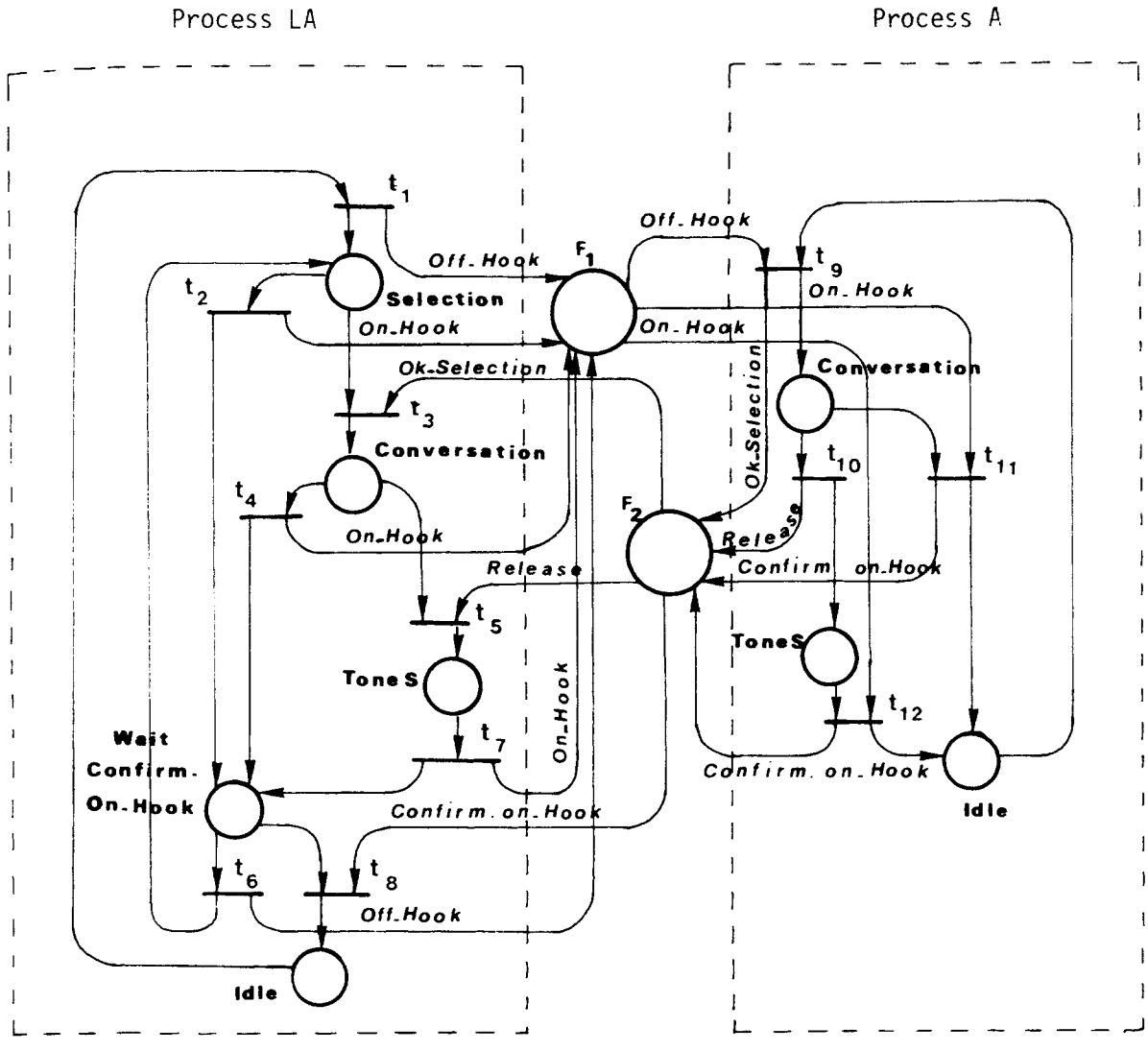


Fig. 2.

The firing of a transition t consists for each fifo f of input of t (i.e., $t \in \Gamma(f)$) to remove $V(f, t)$ which is then prefix of $M(f)$ and for each fifo f' of output of t (i.e., $f' \in \Gamma(t)$) to append $V(t, f)$ to $M(f')$; then $V(t, f')$ is a suffix of $M'(f')$. So, the simulation of FIFO mechanisms becomes quite natural.

Example 1.1. (See Fig. 3). $A = \{\alpha, \beta, \gamma\}$. $M(f_1) = \beta\alpha a$ where $a \in A^*$; $M(f_2) = \gamma\gamma$; $M(f_3) = c$ where $c \in A^*$. t is enabled at M. $M(t\rangle M'$ with $M'(f_1) = a$, $M'(f_2) = \gamma\alpha$, $M'(f_3) = c\gamma\gamma$.

The firing notion is extended to words.

Definition. A *firing sequence* x from a marking M, defined as $x = x_1 \ldots x_k$, is a finite sequence of transitions and:

(i) $x = \lambda$; then x is always defined which is denoted by $M(\lambda\rangle$ and we do not change of marking: $M(\lambda\rangle M$.

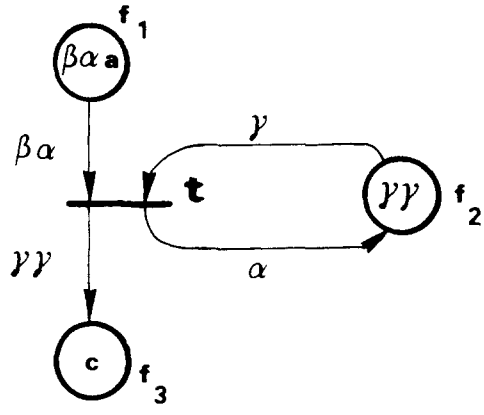


Fig. 3.

(ii) $x = x_1 \dots x_k$; then x is defined iff
- $x_1 \dots x_{k-1}$ is defined; we reach a marking M' from M and write $M(x_1 \dots x_{k-1})M'$,
- $M'(x_k\rangle$ is defined; we reach a marking M'' and write $M'(x_k)M''$.

A marking M' is said to be *reachable* from M if

$$\exists w \in T^*: \quad M(w)M'.$$

This is also denoted by $M(*)M'$.

1.3. Basic properties

As for any transition system, we want to prove that a net has essentially two kinds of properties. On the one hand, a fifo must not accumulate an infinite number of letters (or messages). On the other hand, it is sometimes critical that a transition may not become enabled at any reachable marking. These properties can be checked by analyzing two sets associated with a FIFO net.

Definition. For a FIFO net $N = \langle R ; M_0 ; A\rangle$, the *language of firing sequences* is defined by $F(N) = \{x \in T^* \mid M_0(x\rangle\}$.

The *reachability set* from M_0 is denoted by $\mathrm{Acc}(N) = \{M \mid M_0(*)M\}$.

In concurrent systems, processes can be defined by sets of sequences of actions. In a net, an action may be modelled by several transitions. The firing of some transitions may be either ignored or not observable. Instead of firing sequences, we are more generally interested in sequences of actions. In other words, we label a transition by a letter or by the empty word.

Definition. A *labelled FIFO net* is a couple $N_1 = \langle N ; h\rangle$ where N is a FIFO net, h is a labelling function $h: T \to X \cup \{\lambda\}$, where X is a finite alphabet; h is naturally extended to words.

When $h: T \to X$, we say that the labelling is λ-*free.*

$L(N_1) = \{h(x) \mid x \in F(N)\}$ denotes the *language of the labelled net.*

We find the same properties as for Petri nets.

Definition. A fifo f is *bounded* in a FIFO net $N=\langle R; M_0; A\rangle$ iff $\exists k\in\mathbb{N}$, $\forall M\in \mathrm{Acc}(N): |M(f)|\leq k$.

N is bounded iff each fifo is bounded.

As in Petri nets, a transition t is live if for each reachable marking m there exists a sequence x such that xt is enabled at m.

1.4. A first abbreviation

FIFO nets have been created in order to model and analyze concurrent systems. It is then interesting to increase the descriptive power of the model by adding some abbreviations. In this paper we introduce one abbreviation which is of some usefulness when practicing in modelling sequential processes communication by FIFO channels [13]. This abbreviation consists in modelling a subset of transitions by only one transition under some conditions: if $t_1, \ldots, t_k$ have the same inputs and the same outputs, then a unique transition t can represent them. t is enabled iff there exists a transition t_j enabled. The firing of t then corresponds to the firing of a transition t_j.

Paradigm: Let $Q=\{q_1, \ldots, q_k\}$ be a finite set of A^*. In particular, for decoding the marking of a fifo, subgraphs of the type shown in Fig. 4 are often designed.

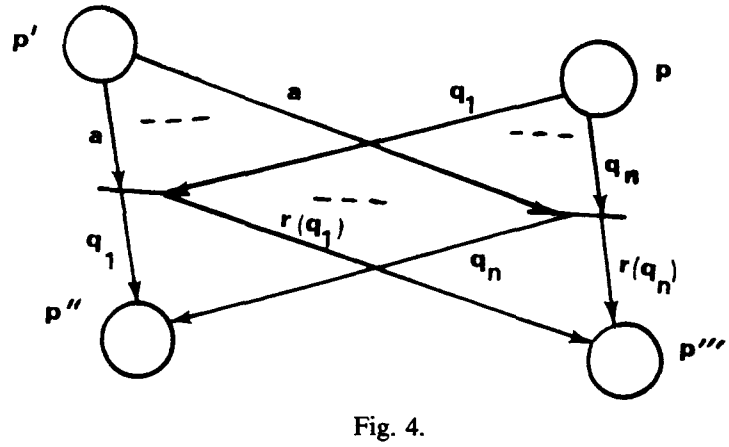


Fig. 4.

Here, $a\in A^*$, $r: Q\to A^*$.

This subgraph is then substituted by the one depicted in Fig. 5.

This kind of folding partially takes into account the ones for coloured Petri nets [9] or for predicate/transition nets [7]. Other types of abbreviation remain to define for FIFO nets.

2. Computational power of the model

We are going to show that FIFO nets have the computational power of Turing machines. First we show how to effectively simulate Petri nets and coloured Petri nets.

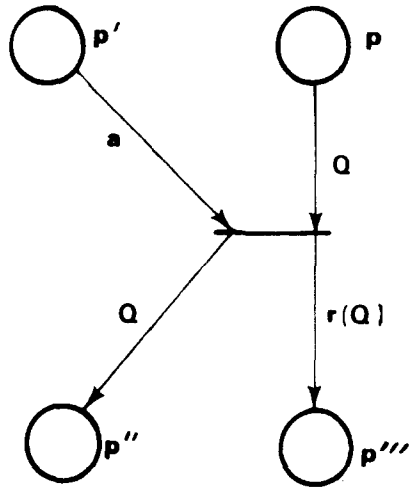


Fig. 5.

2.1. Petri nets

Let $N = \langle\langle P, T ; \Gamma ; V\rangle ; M_0\rangle$ be a Petri net. The simulation of this net by a FIFO net is straightforward. It suffices to construct the FIFO net $\langle\langle P, T ; \Gamma ; V\rangle ; M_0 ; \{m\}\rangle$ where $V(x, y)$ is the word of m^* of length $V(x, y)$. $M(p)$ is the word of length $M(p)$.

2.2. Coloured Petri nets

Coloured Petri nets have been introduced by Jensen [9] or Peterson [15] in order to model complex systems. Here we consider coloured Petri nets without any folding on transition. Let us first present the following definition of coloured nets.

Definition. A *coloured net* is a triplet $\langle R ; M_0 ; C\rangle$ where:

(i) $R = \langle P, T ; \Gamma ; V\rangle$ is a finite valuated bipartite graph where P and T are respectively the sets of places and transitions; Γ is the corresponding between P and T; V is a valuation on $\mathbb{N}^C$ ($\mathbb{N}^C$ is the set of applications from C to $\mathbb{N}$).

(ii) M_0, the initial marking, is a function from P to $\mathbb{N}^C$.

(iii) C is a finite set of *colours*.

Indeed, a transition t is enabled at the marking M iff

$$\forall p \in P,\ \forall c \in C:\quad V(p, t)(c) \leqslant M(p)(c).$$

Firing t, we reach M' verifying

$$\forall p \in P,\ \forall c \in C:\quad M'(p)(c) = M(p)(c) - V(p, t)(c) + V(t, p)(c).$$

We simulate such a net with the following FIFO net:

$$N' = \langle\langle P \cup P_A ; T \cup T_A ; \Gamma' ; V'\rangle ; M_0 ; C\rangle.$$

The alphabet of the net is C. The problem is to ignore the order of the messages in the fifo. Auxiliary fifos and transitions are respectively denoted by P_A and T_A. With each fifo p of P we associate the subgraph shown in Fig. 6, where $p'_p \in P_A$, $a_p, b_p, c_p \in T_A$.

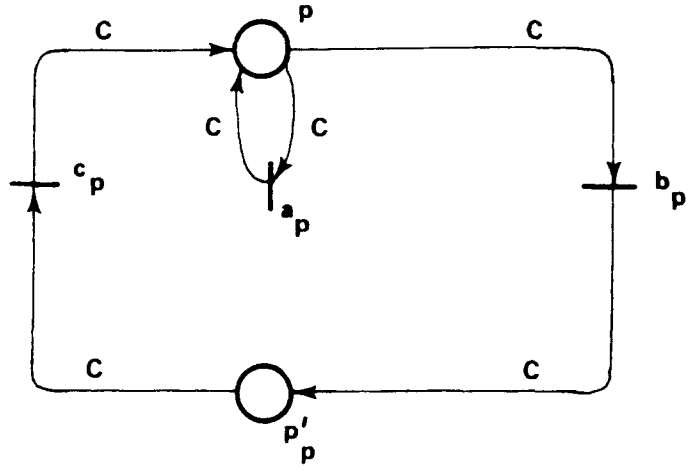


Fig. 6.

Then we have $|P_A| = |P|$ and $|T_A| = 3|P|$. Γ' is defined as follows:

$$\forall p \in P: \quad \Gamma'(p) = \{a_p, b_p\} \cup \Gamma(p),$$

$$\forall t \in T: \quad \Gamma'(t) = \Gamma(t).$$

For a_p, b_p, c_p associated with p we have

$$\Gamma'(a_p) = \Gamma'(c_p) = \{p\}, \qquad \Gamma'(b_p) = \{p'_p\}.$$

For p'_p associated with p we have

$$\Gamma'(p'_p) = \{c_p\}.$$

V' is defined as follows: $\forall x, y \in P \cup T$, $V'(x, y)$ is a word such that the number of occurrences of each letter is exactly the one specified in $V(x, y)$, the Parikh's image of $V'(x, y)$. Thus there are several FIFO nets which can simulate the same coloured net. The valuation of the edges connected to P_A and T_A is described in Fig. 6.

Proposition 2.1. *$F(N) = L(\langle N', h\rangle)$ where h is the following labelling function in $T \cup \{\lambda\}$:*

$$\forall t \in T: \quad h(t) = t,$$

$$\forall t \in T_A: \quad h(t) = \lambda.$$

Sketch of the proof. $\bar{M}_0(p)$ is a word having $M_0(p)$ as Parikh's image. The subgraph described in Fig. 6 allows to generate from $\bar{M}_0(p)$, all the words having $M_0(p)$ as Parikh's image. To prove this, it is enough to number from 1 to n all the tokens in

$M(p)$ (differently coloured or not), and to show that we can generate the permutation group S_n on $\{1, \ldots, n\}$ [13].

So we can use colour as an abbreviation in FIFO nets [13].

2.3. *Alphabetical FIFO nets*

We introduce alphabetical FIFO nets and show how to effectively simulate a FIFO net.

Definition. $\langle R ; M\rangle$ is an *alphabetical* FIFO net iff $\forall (x, y) \in (F \times T) \cup (T \times F)$, $|V(x, y)| \leq 1$.

Each edge is labelled by a letter of A.

We are now going to show how to effectively simulate the behaviour of a FIFO net by an alphabetical one. We proceed by induction on the length l of the longest word valuating an edge of the net.

- If $l = 1$, we have the alphabetical FIFO nets by definition.
- Let us assume that alphabetical FIFO nets can simulate any FIFO net for which $l = L$.
- Let us consider a FIFO net R for which $l = L+1$. It is enough to show that we can simulate N by a FIFO net N' for which $l' \leq L$.

From $N = \langle R ; M_0 ; A\rangle$ we construct $\langle R' ; M_0' ; A'\rangle$ with $A' = A \cup \{[\,,]\}$. [and] being two letters which do not exist in A, will be used to parenthesize the marking of any fifo of R in $R' = \langle F \cup F_A ; T \cup T_A ; \Gamma' ; V'\rangle$.

M_0' is defined as follows:

$$\forall f \in F: \quad M_0'(f) = [M_0(f)]],$$

$$\forall f \in F_A: \quad M_0'(f) = \lambda.$$

The transformation from R to R' consists in substituting for each transition t the following subgraphs. Without loss of generality let us take the situation, depicted in Fig. 7, where d, b, γ, $\beta \in A^{L+1}$ and $d = d_1 \ldots d_{L+1}$.

We show the transformation of t by the four subgraphs, shown in Figs. 8 to 11, respectively. The three first ones decode the marking of B, C, D and say when t is enabled in N.

The last one either restores the markings of B, C, D if t is not fired, or simulates the firing of t in N.

The decoding of D is similar to the decoding of B. Fifos R_x store the decoded part of the marking of x. $\neq^1$ and $\neq^L$ are abbreviations for any letter different of x_{L+1} and any word of length less than or equal to L different from $x_1 \ldots x_L$. If $d \leq M(D)$, then p_d is marked, else $p_{\bar{d}}$ is marked (see Fig. 9). Then t is enabled at M in N iff 'test' becomes enabled in N' after the decoding of M'. Either t is fired and

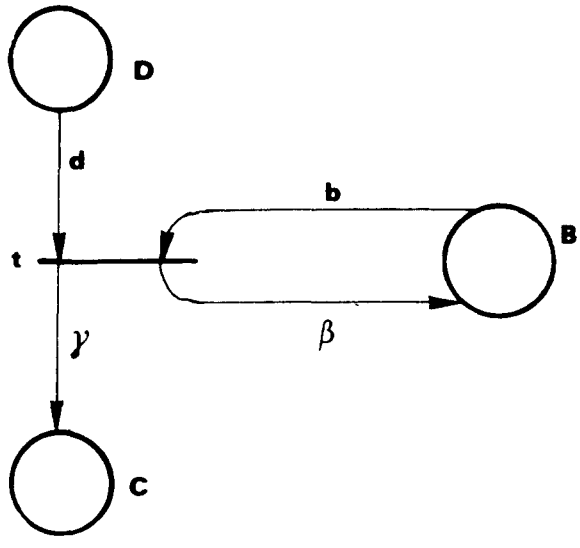


Fig. 7.

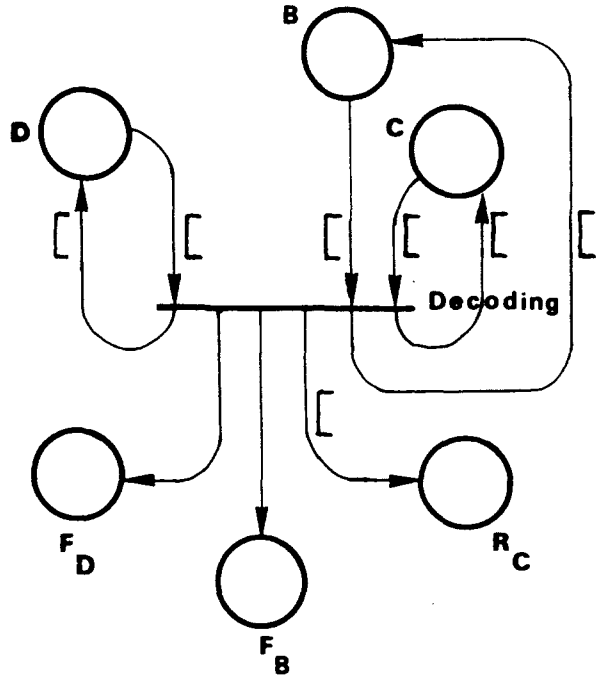


Fig. 8.

then fifos S_x are marked, or we must restore M'. We construct the end of the automaton, as shown in Fig. 11 for the fifo B.

Let us point out that R' is such that $l' \leq L$. If, for instance, $M(B) = x$ in N, and we have $M'(B) = [x]]$ in N', let us show how the marking of B evolves in N'.

After the firing of 'Decoding', we have $x]][$. Let $x_1\ x_2$ be such that $x] = x_1\ x_2$ and before the firing of 'test', we have $x_2][$ in B and x_1 in R_B. If we fire 'test', then $x_1 = b$, R_B contains $b[$ and S_B is marked. We cancel all the letters in R_B, then permute the letters in B which reach $]][x_3$ where $x_2 = x_3]$.

We then put β in B, permute the last] which locks any other automaton associated with B and another transition of T. Then we find $[x_3\ \beta]]$ in B which corresponds to $[(\delta_t(M))(b)]]$ where δ_t is the firing function of t in N. If E_B is marked, then

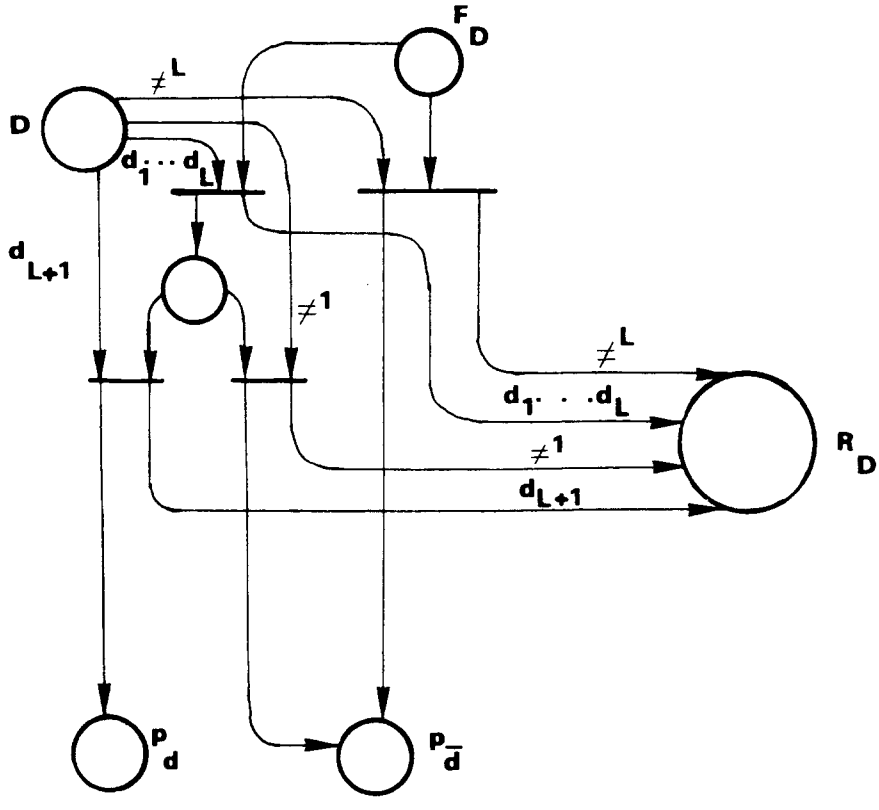

F D
≠ L
D
d 1 .. d L
d L+1
≠ 1
≠ L
d 1 . . . d L
≠ 1
d L+1
R D
p d
p d̄

Fig. 9.

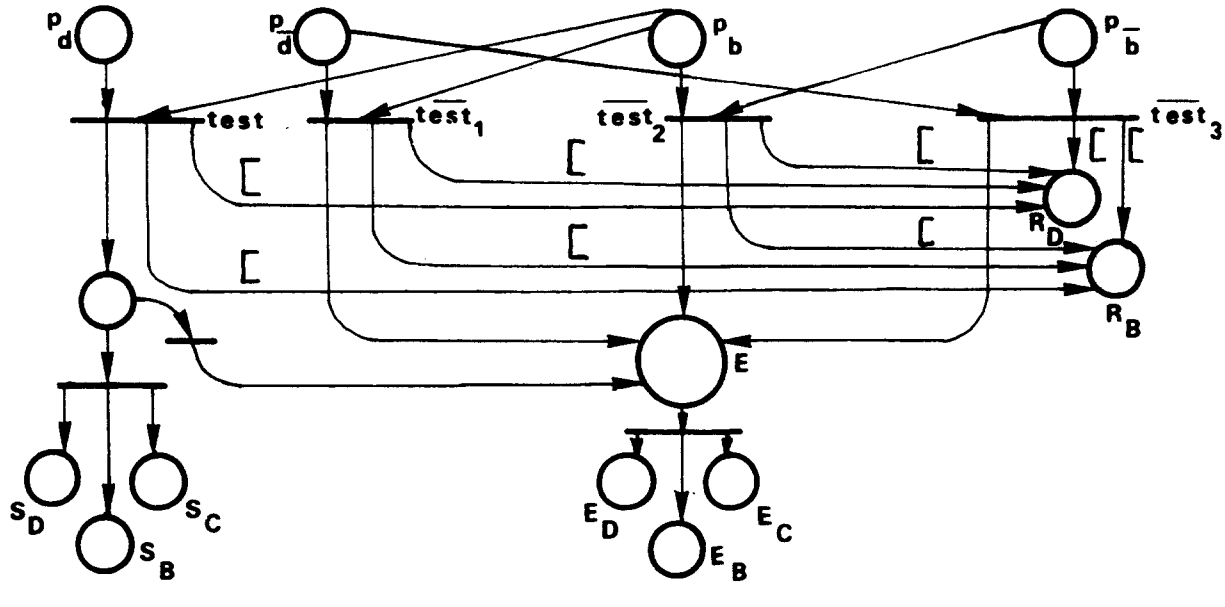

p d
p d̄
p b
p b̄
test
test 1
test 2
test 3
R D
R B
E
S D
S C
S B
E D
E C
E B

Fig. 10.

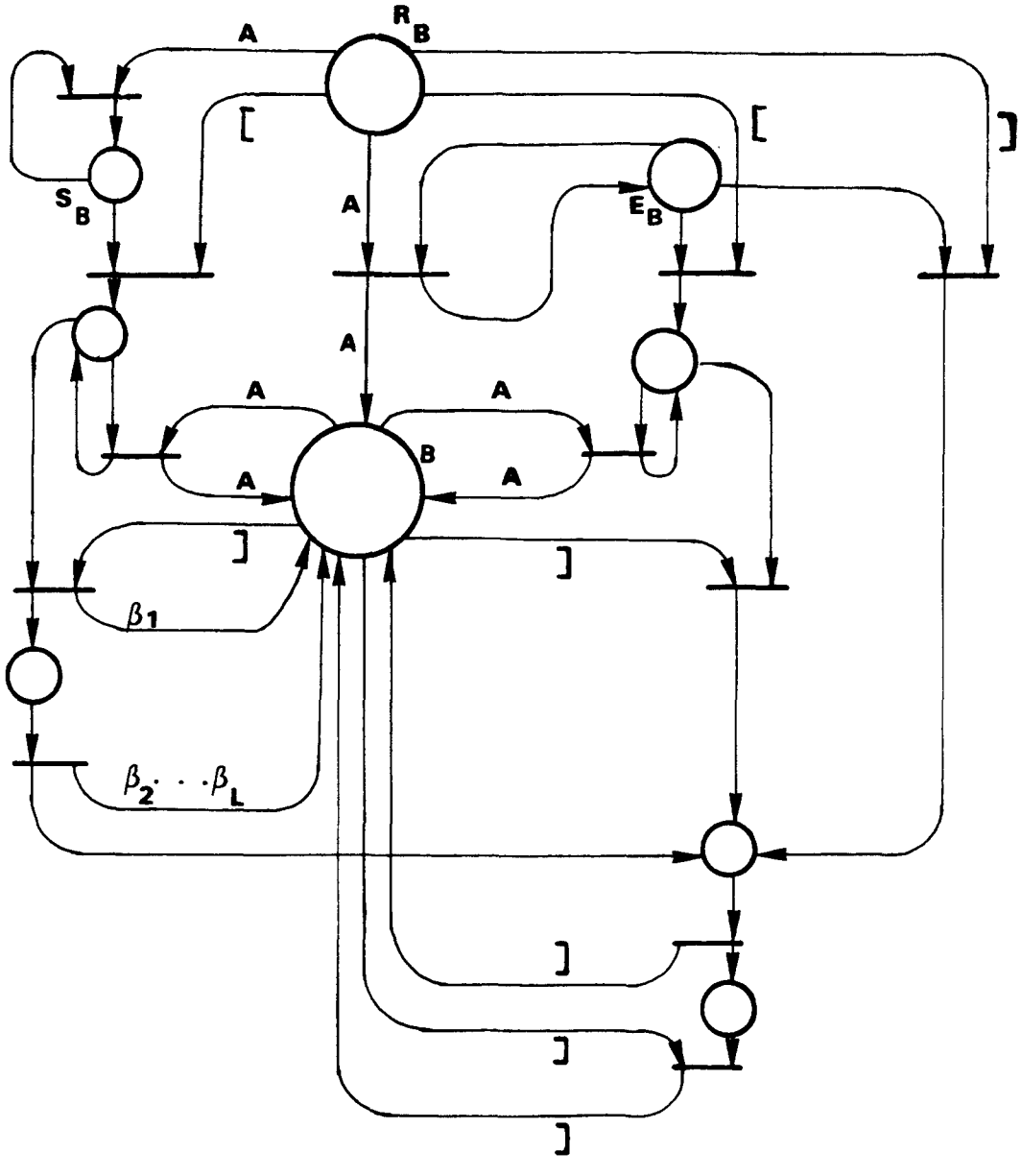


Fig. 11.

we have to restore the marking of B by moving the letters from R_B to B. Then the marking of B is $x_2][\ x_1$. We permute the letters of x_2 and]. Finally, we find $[x]]$.

We have the following result.

Theorem 2.2. $L(\langle R\,;M_0\,;A\rangle)=L(\langle R'\,;M_0'\,;A'\rangle,h)$ *with the following labelling function*:

$$\forall t\in T:\quad h(t)=t,$$

$$\forall t\in T_A:\quad h(t)=\lambda.$$

2.4. Program machine

We show that a class of FIFO nets has the power of the class of Turing machines, by simulating a program machine [14, 21].

Each program machine is defined by finite sets of registers of instructions labelled by a different label for each instruction. There are two kinds of instructions: the increment of a register, and the test and decrement of a register.

We simulate a program machine by an alphabetical FIFO net $\langle R ; M_0 ; A \rangle$ such that $A = \{0, 1\}$. Each register r_i is associated with a fifo r_i.

An integer n is coded by the word

$$0 \underbrace{1 \ldots 1}_{n \text{ times}} = 0\ 1^n\ 0.$$

Each label q_i is associated with a fifo q_i.

The increment of a register ($q_i : r_j := r_j + 1$ **goto** q_m ;) is simulated by the subgraph, shown in Fig. 12. r_j contains $0\ 1^n\ 0$. The firing of t_{i_1} is the beginning of the instruction. r_j contains now $1^n\ 00$; then we fire t_{i_2} n times which permutes the 1: r_j contains now $00\ 1^n$; t_{i_3} is then enabled and adds one 1 in r_j; t_{i_4} ends the instruction suffixing r_j by 0.

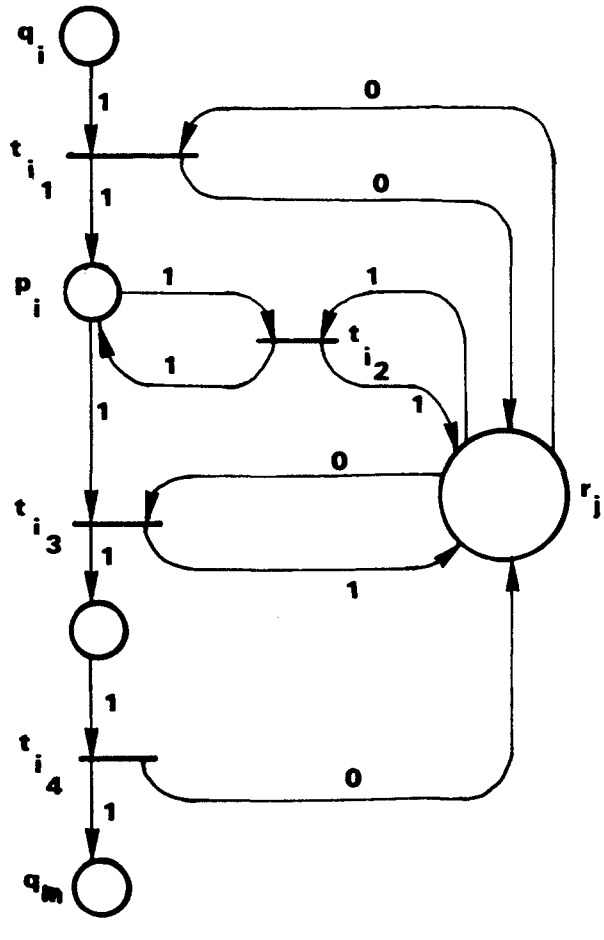


Fig. 12.

The test and decrement instruction (q_i: **if** $r_j := 0$ **then goto** q_m **else** $r_j := r_j - 1$; **goto** q_1 **endif**) is simulated by the subgraph, shown in Fig. 13. r_j contains 01^n0. The firing t_{i_1} is the beginning of the instruction, r_j contains now 1^n00; t_{i_2} tests if $r_j = 0$; if $r_j \neq 0$, then t_{i_3} is the only transition enabled. Then t_{i_4} permutes all the 1's.

After t_{i_5} fires, r_j contains $01^{n-1}0$.

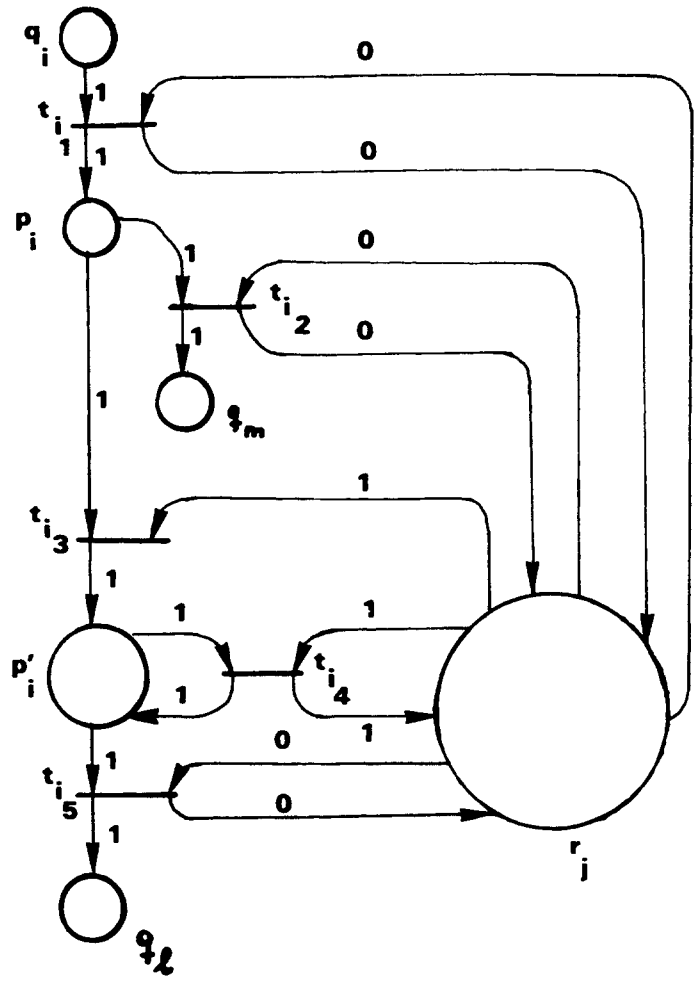


Fig. 13.

Fifos q_i and auxiliary fifo p_i and p_i' are the support of a state machine S.

The initial marking M_0 is defined by $M_0(q_0) = 1, \forall f \in S - \{q_0\}; M_0(f) = \lambda ; M_0(r_i) = 01^n0$, if r_i contains n in the program machine.

From these two subgraphs we have the following theorem.

Theorem 2.3. *Alphabetical FIFO nets have the computational power of Turing machines.*

It is enough to invoke Church's thesis to deduce that FIFO nets have the power of Turing machines. Nevertheless, Theorem 2.2, showing an effective means to simulate FIFO nets, gives us a proof of the equivalence of the two models. Another method [4] consists of showing that the family of the languages associated with the labelled FIFO nets is closed under rational transduction and contains a generator (the anti-Dyck) of the family of the r.e. languages [20].

These last results lead us to define sub-classes of FIFO nets where boundedness and liveness would be decidable.

3. Monogeneous FIFO nets

Our aim is to construct a class of FIFO nets containing at least a class isomorphic to Petri nets, and for which the boundedness problem is decidable. The procedure of Karp and Miller [10], used for Petri nets, is mainly based upon two considerations:

(1) If $M \geqslant M'$, then $\mathrm{Acc}(\langle R ; M'\rangle) \subseteq \mathrm{Acc}(\langle R ; M\rangle)$, i.e., if a marking M is greater than or equal to M', then each sequence enabled at M' is also enabled at M.

(2) Koenig-Duckson Lemma: From each infinite sequence of vectors (here of markings) in $(\mathbb{N} \cup \{\omega\})^r$ one can extract a nondecreasing subsequence.

First, we define an order on a set of markings as large as possible. This relation satisfies the two considerations we mentioned. We then convince ourselves that the valuation of the input edges of a fifo has to be restricted. Otherwise, we could not be sure that the reachability set of a net would be contained in the set in which our order is defined. These reflexions led us to the definition of 'monogeneous FIFO nets'. More precisely, a FIFO net is monogeneous if each input edge of a fifo is valuated by a power of the same word.

3.1. Boundedness is decidable for monogeneous FIFO nets

Definition. A FIFO net $\langle\langle F, T, \Gamma, V\rangle ; M_0\rangle$ is a *monogeneous FIFO net* iff

$$(\forall f_i \in F)(\exists u_i \in A^*)(\forall t \in T)(V(t, f_i) \in u_i^*).$$

A word u of A^+ is *primitive* iff $\forall v \in A^+, \forall n \geqslant 2, u \neq v^n$. We denote by u_i the unique primitive word associated with the fifo f_i.

Theorem 3.1. *Let $N = \langle R ; M_0\rangle$ be a monogeneous FIFO net. Then the boundedness of* $\mathrm{Acc}(N)$ *is a decidable problem.*

Proof. We define an order relation, denoted $\succcurlyeq$, on

$$E = \left\{ \prod_{i=1}^{|F|} (\mathrm{suffix}\,(u_i \cup M_0(f_i)))u_i^{\infty} \right\}$$

such that

$$\forall M, M' \in E,\ M' \succcurlyeq M \ \Leftrightarrow\ \forall i = 1, \ldots, |F|,\ M'(f_i) \in M(f_i)u_i^{\infty}.$$

This ordering satisfies our considerations (1) and (2) in $(E, \succcurlyeq)$. For a monogeneous FIFO net, $\mathrm{Acc}(N) \subseteq E$. Then "$\mathrm{Acc}(N)$ is infinite" is equivalent to "there exists a firing sequence x such that if $M(x\rangle M'$ with $M' \succcurlyeq M$ and $M \neq M'$."

This equivalence allows us to extend the construction of the coverability tree [10, 21] from Petri nets to monogeneous FIFO nets.

The coverability tree $\mathrm{CT}(N)$ of a monogeneous net $N = \langle R ; M_0\rangle$ is defined by $\mathrm{CT}(N) = \langle S, X\rangle$ where S is a set of nodes labelled by elements of E, and X is a set of edges labelled by elements of T.

CT(N) is defined by the following conditions:

(1) The root r is labelled by M_0.

(2) If s is a node labelled by Q, then s has no successor when either

(a) on a path from r to s there is a node $s_1 \neq s$, also labelled by Q, or

(b) there is no transition t such that $Q \geqslant V(\cdot, t)$.

(3) If s is labelled by Q and s does not satisfy conditions (a) or (b), then for each t of T such that $Q \geqslant V(\cdot, t)$ there is a successor s' labelled by Q' with

(i) $Q'(f_i) = Q_1(f_i)u_i^{\omega}$ for any fifo f_i of F, for which there is a node s_2 on the path from r to s (including s) labelled by Q_2 with $Q_1 \gg Q_2$ and $Q_2(f_i) \neq Q_1(f_i)$.

(ii) $Q'(f_i) = Q_1(f_i)$ otherwise, where $Q_1(f_i) = g(V(f_i, t), Q(f_i)) \cdot V(t, f_i)$.

The edge from s to s' is labelled by t.

Assuming the contrary and applying the Koenig–Duckson Lemma to $(E, \gg)$ we prove that CT(N) is finite. We show, with the definition of CT(N), that "Acc(N) is infinite" is equivalent to "there exists a node s in CT(N) labelled by Q such that $|Q(f_i)| = \omega$ for a fifo f_i of F."

Thus we have found an effective procedure to decide the boundedness of Acc(N).

Example 3.2. $N = \langle R ; M_0 \rangle$; $M_0 = (b, a)$ (see Fig. 14 and the coverability tree shown in Fig. 15).

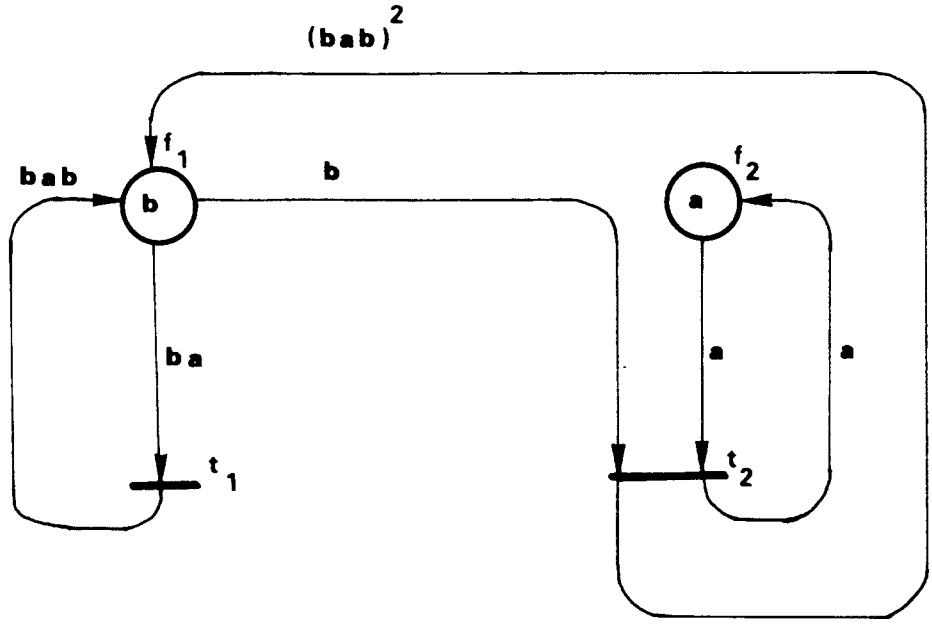


Fig. 14.

We can also extend the definition of a coverability graph [21, 3], which we recall here.

Definition. The *coverability graph* CG(N) of a net N is obtained from the coverability tree CT(N) by identifying the nodes having the same label.

The coverability graph of the net of Example 3.2 is the one depicted in Fig. 16.

CT(N)

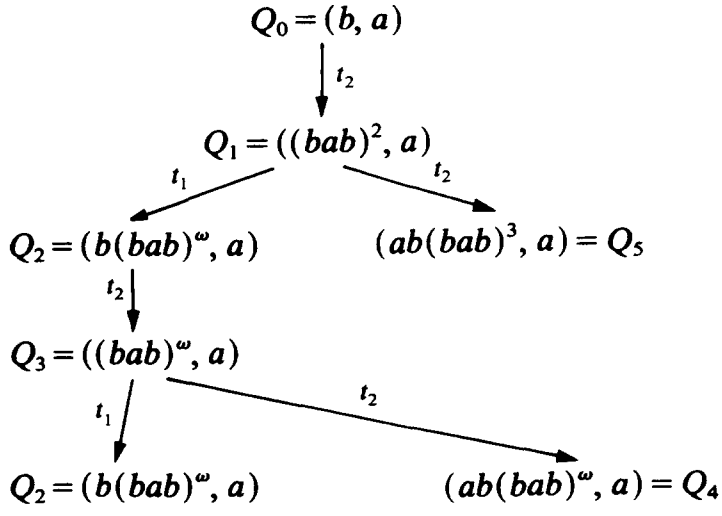


Fig. 15.

CG(N)

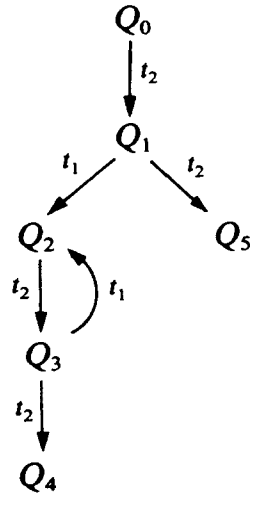


Fig. 16.

Notation. Let $Q \in (A^\infty)^F$ and $M \in (A^*)^F$.

$$Q^{-1}(\omega) = \{f_i \in F \mid |Q(f_i)| = \omega\},$$

$$Q \overset{\text{fin}}{=} M \Leftrightarrow \forall f_i \notin Q^{-1}(\omega)\colon Q(f_i) = M(f_i).$$

An edge from Q to Q' labelled by t is written $(Q \xrightarrow{t} Q')$. A sequence of labels on the edges of a path is called the *word* of the path. The theorem below justifies the name 'coverability tree'. As a matter of fact, for each marking M of Acc(M_0) there exists a node of CG(N) (then of CT(N)), labelled by Q, which covers M (i.e., such that $Q \geqslant M$).

Theorem 3.3. *Let $N=\langle R; M_0\rangle$ be a monogeneous net and* CG(N) *its coverability graph.*

(a) *If $M_0(u\rangle M$, then there exists a unique path in* CG(N) *labelled by u from M_0 to a node labelled by Q such that $Q \geqslant M$ and $Q \overset{\text{fin}}{=} M$.*

(b) *For each node s of* CG(N) *labelled by Q, for each k of $\mathbb{N}$ there exists M of* Acc(M_0) *such that $Q \overset{\text{fin}}{=} M$ and*

$$\forall f \in Q^{-1}(\omega),\ M(f) \geqslant h(Q(f))u_f^k \quad \text{with } h(v_i u_i^n) = v_i,$$
$$\forall i = 1, \ldots, p,$$
$$v_i \notin A^* u_i u_i^* \ (p = \text{card } F).$$

Proof. (a) (by induction on $|u|$). It is obviously true for $|u|=0$ iff $u=\lambda$.

Let $u=vt$ with $v\in T^*$, $t\in T$ and $M_0(v\rangle M_1$. Let Q_1 be the label of the node s_1 such that $M_0 \xrightarrow{v} Q_1$. Then, by induction hypothesis, $Q_1 \overset{\text{fin}}{=} M_1$ and $Q_1 \geqslant M_1$ and $M_0(vt\rangle$ implies $M_1 \geqslant V(\cdot, t)$. Then there exists, in CT(N), a node s_2 labelled by Q_2 such that $Q_2 = g(V(\cdot, t), Q_1)\cdot V(t, \cdot)$ and $s_1 \xrightarrow{t} s_2$. Let $M_1(t\rangle M_2$. We have $M_2 = g(V(\cdot, t), M_1)\cdot V(t, \cdot)$, and $Q_1 \geqslant M_1$ implies $Q_2 \geqslant M_2$ and $Q_2 \overset{\text{fin}}{=} M_2$.

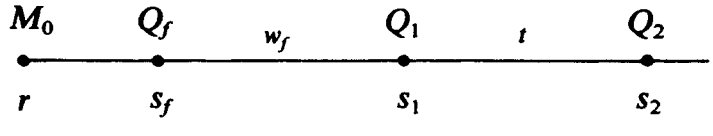
(b) (by induction on the depth of the coverability tree). The property obviously holds for root r labelled by M_0. Let us assume it is true for node s_1 labelled by Q_1. Let s_2 be a successor of s_1 labelled by Q_2 and such that $s_1 \xrightarrow{t} s_2$.

We have $Q_1^{-1}(\omega) \subseteq Q_2^{-1}(\omega)$.

Let $f \in Q_2^{-1}(\omega) - Q_1^{-1}(\omega)$. Then there exists, on the path from r to s_1, a node s_f labelled by Q_f such that:

$$\begin{cases} Q_f \lessdot g(V(\cdot, t), Q_1)\cdot V(t, \cdot), \\ Q_f(f) \neq g(V(f, t), Q_1(f))\cdot V(t, f). \end{cases}$$

Let w_f be the word of T^* labelling the path from s_f to s_1.



We have $(\forall f' \notin Q_2^{-1}(\omega))$:

$$Q_f(f') = Q_2(f'), \quad Q_f \lessdot Q_2 \quad \text{and} \quad Q_f(f') \neq Q_2(f').$$

Let us denote

$$Q_1^{-1}(\omega) - Q_2^{-1}(\omega) = \{f_1, \ldots, f_m\},$$
$$w_{f_i} = w_i \quad \forall i = 1, \ldots, m.$$

Let

$$K' = K + K\cdot\left(\sum_{i=1}^{m} k_i\right) + \left(\sup_{f\in Q_1^{-1}(\omega)} |V(f, t)|\right)$$

with

$$k_i = \sup_{f \in Q_1^{-1}(\omega)} |V(f, w_i t)| \quad \forall i = 1, \ldots, m.$$

By hypothesis there exists an M_1 in $\mathrm{Acc}(N)$ such that

$$\begin{cases} M_1(f) = Q_1(f) & \forall f \notin Q_1^{-1}(\omega), \\ M_1(f) \geqslant h(Q_1(f)) \cdot u_f^{K'} & \forall f \in Q_1^{-1}(\omega). \end{cases}$$

Then

$$\begin{cases} M_2(f) = Q_2(f) & \forall f \notin Q_2^{-1}(\omega), \\ M_2(f) \geqslant h(Q_2(f)) \cdot u_f^{K} & \forall f \in Q_2^{-1}(\omega). \end{cases}$$

We can now make precise Theorem 3.1 by showing that we can decide, with the coverability graph, whether a fifo of the net is bounded or not.

Theorem 3.4. *Let $N = \langle R ; M_0 \rangle$ be a monogeneous net. The fifo f_i is unbounded iff there exists a node Q in $\mathrm{CG}(N)$ such that $|Q(f_i)| = \omega$.*

Proof. (1) f_i is unbounded iff $(\forall k \in \mathbb{N})(\exists M_k \in \mathrm{Acc}(N))(|M_k(f_i)| > k)$. As the coverability graph contains a finite number of vectors of $(A^\infty)^{\mathrm{card}\, F}$, Theorem 3.3(a) implies that there exists a vector Q such that $|Q(f_i)| = \omega$.

(2) Conversely, if there exist a vector Q of $\mathrm{CG}(N)$ and a fifo f_i in R such that $|Q(f_i)| = \omega$, then Theorem 3.3(b) implies f_i is not bounded.

We extend a theorem from [21] as follows.

Theorem 3.5. *Let $N = \langle R ; M_0 \rangle$ be a monogeneous net and $\mathrm{CG}(N)$ its coverability graph.*

(a) *If (Q_1, t, Q_2) is an edge in $\mathrm{CG}(N)$, then $Q_1 \geqslant V(\cdot, t)$ and for each f_i of F, $Q_2(f_i)$ is $Q_1(f_i) \cdot u_i^\omega$ or $g(V(f_i, t), Q_1(f_i)) \cdot V(t, f_i)$. In particular, $Q_1^{-1}(\omega) \subseteq Q_2^{-1}(\omega)$.*

(b) *Let $v \in (A^*)^{\mathrm{card}\, F}$; then*

$$\exists M \in \mathrm{Acc}(N),\ M \geqslant v \text{ iff } \exists Q \in \mathrm{CG}(N),\ Q \geqslant v.$$

(c) *$F(N)$ is infinite iff there exists a circuit in $\mathrm{CG}(N)$.*

Proof. (a) is obvious.

(b) is straightforward from Theorem 3.3.

Let us examine (c). $F(N)$ infinite, $\mathrm{CG}(N)$ finite and each word of $F(N)$ labelling a path in $\mathrm{CG}(N)$ imply there is a circuit in $\mathrm{CG}(N)$.

Conversely, let $Q_1, \ldots, Q_s = Q_1$ be a circuit labelled by v in $\mathrm{CG}(N)$. There are two cases:

(1) $\exists i \in \{1, \ldots, s-1\}$, $Q_i^{-1}(\omega) \neq \emptyset$. Then, $F(N)$ is infinite.

(2) $\forall i \in \{1, \ldots, s-1\}$, $Q_i^{-1}(\omega) = \emptyset$. Then $Q_1, \ldots, Q_s \in \mathrm{Acc}(N)$, and therefore $\exists u$, $v \in T^*$ such that $M_0(u\rangle Q_1(v\rangle Q_1$. So $uv^* \subseteq F(N)$; $|F(N)| = \omega$.

We show that monogeneous nets are included in deterministic Petri nets with λ-free labelling functions. So, we can derive that regularity is decidable in the set of monogeneous net languages.

3.2. Regularity in F_m is decidable

Definition (Vidal-Naquet [22]). A labelled monogeneous (or Petri) net $\langle N, h\rangle$ is *deterministic* iff $h: T \to X \cup \{\lambda\}$ satisfies the following:

$$\forall M \in \mathrm{Acc}(N), \forall t, t' \in T, \quad (M(t\rangle \text{ and } M(t'\rangle \text{ implies } t = t' \text{ or } h(t) \neq h(t')).$$

In [18], this notion is a basic hypothesis and is known as the 'disjoint labelling condition'.

Property 3.6. *Let $\langle R, M_0, h\rangle$ be a deterministic labelled FIFO net such that $h: T \to X$. Then for each x of X^* there exists at most one marking M and one word u of T^* such that $M_0(u\rangle M$ with $h(u) = x$.*

Proof (by induction on the length of u).

The classes of languages F_m, D_m^c, L_m and Pref(Reg) are defined as follows:

$F_m = \{F(N) \mid N \text{ is a monogeneous net}\}$,
$D_m^c = \{h(F(N)) \mid \langle N, h\rangle \text{ is a deterministic monogeneous net and } h: T \to X\}$,
$L_m = \{h(F(N)) \mid \langle N, h\rangle \text{ is a labelled monogeneous net}\}$,
$L_m^c = \{h(F(N)) \mid \langle N, h\rangle \text{ is a labelled monogeneous net and } h: T \to X\}$.

We define in a similar way the set of languages of Petri nets that we denote F_p, D_p^c, L_p and L_p^c as in [22]. Pref(Reg) is the set of the regular language L such that $\mathrm{Pref}(L) = L$.

We show that the language $F(N)$ of a monogeneous net N is equal to the intersection of $F(\bar{N})$ and of $L(A_N)$ where $\bar{N}$ is the Petri net associated with N and $L(A_N)$ is a regular language associated with $\mathrm{CG}(N)$.

Definition. Let $N = \langle R, M_0\rangle$ be a monogeneous net. We denote by:

(1) A_N the finite automaton $A_N = \langle X, E, e_0, \delta, E_f\rangle$ with $X = T$, $e_0 = M_0$, $E = E_f$. The set of nodes of $\mathrm{CG}(N)$ is E. $\delta(e, t) = e'$ iff $e \xrightarrow{t} e'$.

(2) $\bar{N}$ is the Petri net $\langle \bar{R}, \bar{M}_0\rangle$ where

$$\bar{R} = \langle F, T, \Gamma, \bar{V}\rangle, \qquad \bar{V}(x, y) = |V(x, y)|, \qquad \bar{M}_0(f) = |M_0(f)| \ \ \forall f \in F.$$

Theorem 3.7. *Let N be a monogeneous net; then*

$$F(N) = F(\bar{N}) \cap L(A_N).$$

Proof. Obviously, we have

$$F(N) \subseteq F(\bar{N}), \qquad F(N) \subseteq L(A_N);$$

then $F(N) \subseteq F(\bar{N}) \cap L(A_N)$.

We need to show that $F(\bar{N}) \cap L(A_N) \subseteq F(N)$. This is done by induction on the length of a word u of $F(\bar{N}) \cap L(A_N)$:

- $|u| = 1$, so $u \in L(A_N)$ implies $u \in F(N)$,
- $u = vt$, $v \in T^*$, $t \in T$.

We assume $v \in F(N)$ and $vt \in F(\bar{N}) \cap L(A_N)$.

$M_0(v\rangle M$; then there exists a path in CG(N) labelled by v from M_0 to a node Q such that

$$Q \geqslant M \quad \text{(Theorem 3.3(a))}. \tag{1}$$

As $vt \in L(A_N)$ we have

$$\forall f \in F, \quad V(f, t) \leqslant Q(f) \tag{2}$$

and $vt \in F(\bar{N})$; then $\bar{M}(t\rangle$, i.e.,

$$\forall f \in F, \quad \bar{M}(f) \geqslant |V(f, t)|. \tag{3}$$

With (1), (2) and (3) we conclude

$$\forall f \in F, \quad V(f, t) \leqslant M(f).$$

Then $M(t\rangle$ and $M_0(v\rangle M(t\rangle$, $vt \in F(N)$, and then $F(\bar{N}) \cap L(A_N) \subseteq F(N)$.

From Theorem 3.7 we can deduce the following theorem.

Theorem 3.8. (1) $F_p \subsetneq F_m \subsetneq D_p^c$.
(2) $L_m^c = L_p^c$.
(3) $L_m = L_p$.

Proof. (1) $F_p \subseteq F_m$ is obvious.

The language Pref$(abba)^* \in F_m - F_p$ [5], then $F_p \subsetneq F_m$. From Theorem 3.7 we have $F(N) = F(\bar{N}) \cap L(A_N)$.

$$\begin{cases} F(\bar{N}) \in F_p \subseteq D_p^c, \\ L(A_N) \in \text{Pref(Reg)}. \end{cases}$$

Vidal-Naquet [22] has shown that Pref(Reg) $\subseteq D_p^c$. It is easy to see that D_p^c is closed by intersection.

Hence, $F(N) \in D_p^c$, i.e., $F_m \subseteq D_p^c$.

(2) and (3) are obvious by Theorem 3.7 and using the fact that L_p^c and L_p are closed by intersection.

Theorem 3.7 is the key point of this section. In fact, it allows us to decide for the monogeneous nets all the questions that are decidable for Petri nets. Language theory allows to refine comparison between Petri nets and monogeneous nets. In this way, Theorem 3.8 is more precise than a result in [19].

Theorem 3.9. *Regularity is decidable in* F_m.

Proof. The regularity in F_p is decidable [21]; Property 3.6 implies that regularity in D_p^c is also decidable.

As $F_m \subseteq D_P^c$, we conclude that we can decide whether $F(N)$ is regular or not.

Remark. $F(\bar{N}) \in$ Reg implies $F(N) \in$ Reg. The converse is false.

4. Conclusion

We have introduced a novel model of parallel computation. This model has been specially suited in order to design and analyze sequential processes communicating by FIFO channels.

The alphabetical FIFO nets have the same computational power as the Turing machines. We have restricted the valuation of a net and constructed the monogeneous FIFO nets. For this last class, which properly contains the Petri nets, we have extended the procedure of Karp and Miller. Theorem 3.7, showing that the language of a monogeneous FIFO net can be seen as the intersection of a regular language and the language of a Petri net, is the key-point for solving questions of decidability. Finally, the proper inclusion of F_p into F_m justifies the introduction of the monogeneous FIFO nets from a theoretical point of view.

The classes we have dealt with are defined by restriction on the valuation. We are studying classes of FIFO nets defined like Petri nets [3] by restrictions on the structure of the graph of the FIFO net [4].

From a practical point of view, we have released a software tool [1] for analyzing specifications of concurrent systems described in a specification language [2]. This tool uses an algorithm for finding all the minimal semiflows in a FIFO net. In fact, the invariant method [9] can still be applied for FIFO nets [13]. Another procedure of this tool allows us to construct the reachability graph of a FIFO net in the case where it is provided by the translation of a program written in our specification language.

Acknowledgment

The authors wish to thank Professors G. Roucairol and G. Vidal-Naquet for valuable discussions about the problems exposed here. G. Hahn and J. Bond contributed helpful suggestions for the English translation of the manuscript.

References

[1] P. Behm and G. Memmi, RAFAEL, un outil d'analyse de systèmes temps réel, *2ème Coll. de Génie Logiciel AFCET*, Nice, 1984.

[2] F. Boussinot, R. Martin, G. Memmi, G. Ruggiu and J. Vapne, A language for formal description of real time systems, in: J. A. Baylis, ed., *SAFE COMP'83, Proc. 3rd IFAC/IFIP Workshop*, Cambridge, England (Pergamon Press, New York, 1983) 119-126.

[3] G.W. Brams, *Réseaux de Petri: Théorie et Pratique Tome 1* (Masson, Paris, 1982).

[4] A. Finkel, Deux classes de Réseaux à files: les Réseaux Monogènes et les réseaux Préfixes, Thèse de 3ème cycle, Rapport LITP No. 83-3 Paris VII, 1982.

[5] A. Finkel, Control of a Petri net by a finite automaton, *3rd Conf. on Foundations of Software Technology and Theoretical Computer Science*, Bangalore, India, 1983.

[6] A. Finkel and G. Memmi, FIFO nets: A new model of parallel computation, *Proc. 6th G.I. Conf. on Theoretical Computer Science*, Dortmund, Lecture Notes in Computer Science **145** (Springer, Berlin, 1983).

[7] H.J. Genrich and K. Lautenbach, The analysis of distributed systems by means of predicate/transition nets, in: G. Kahn, ed., *Semantics of Concurrent Computation*, Lecture Notes in Computer Science **70** (Springer, Berlin, 1979) 123-146.

[8] C.A.R. Hoare, Communicating sequential processes, *Comm. ACM* **21** (8) (1978) 666-677.

[9] K. Jensen, Coloured Petri nets and the invariant method, *Theoret. Comput. Sci.* **14** (3) (1981) 317-336.

[10] R.M. Karp and R.E. Miller, Parallel program schemata, *J. Comput. System Sci.* **4** (1969) 167-195.

[11] T. Kasai and R.E. Miller, Homomorphisms between models of parallel computation, *J. Comput. System Sci.* **25** (1982) 285-331.

[12] R. Martin and G. Memmi, Specification and validation of sequential processes communicating by FIFO channels, *4th Internat. Conf. on Software Engineering for Telecommunication Switching Systems* (IEE, Warwick, England, 1981).

[13] G. Memmi, Methode d'analyse de réseaux de Pétri, réseaux à files, et applications aux systèmes temps réel, Thèse de Doctorat d'État, Université Pierre et Marie Curie Paris, 1983.

[14] M. Minsky, *Computation: Finite and Infinite Machines* (Prentice-Hall, Englewood Cliffs, NJ, 1967).

[15] J.L. Peterson, A note on colored Petri nets, *Inform. Process. Lett.* **11** (1) (1980) 40-43.

[16] W. Reisig, Deterministic buffer synchronization of sequential processes, *Acta Informatica* **18** (1982) 117-134.

[17] G. Roucairol, Mots de synchronisation, *RAIRO Informatique* **12** (4) (1978) 227-290.

[18] G. Rozenberg and R. Verraedt, Subset languages of Petri nets, *Proc. 3rd European Workshop on Applications and Theory of Petri nets*, Varenna, Italy, 1982; also in: A. Pagnoni and G. Rozenberg, eds., *Informatik-Fachberichte 66*, (Springer, Berlin, 1983) 250-263.

[19] P. Starke, Monogeneous FIFO nets and Petri nets are equivalent, *Bulletin l'EATCS* **21** (1983).

[20] B. Vauquelin and P. Franchi-Zannettacci, Automates à files, *Theoret. Comput. Sci.* **11** (1980) 221-225.

[21] R. Valk and G. Vidal-Naquet, Petri nets and regular languages, *J. Comput. System Sci.* **23** (3) (1981) 229-325.

[22] G. Vidal-Naquet, Rationalité et déterminisme dans les réseaux de Pétri, Thèse d'Etat, Paris VI, 1981.